

































































































































































































































































































































































































































































































































































































```
cout << "Attachment is sending ~~~~~" << endl;
cout << "Please be patient!" << endl;
string sendBuff;
sendBuff = "--qwertyuiop\r\n";
sendBuff += "Content-Type: application/octet-stream;\r\n";
sendBuff += " name = \";
sendBuff += (* pIter) -> fileName;
sendBuff += "\"";
sendBuff += "\r\n";

sendBuff += "Content-Transfer-Encoding: base64\r\n";
sendBuff += "Content-Disposition: attachment;\r\n";
sendBuff += " filename = \";
sendBuff += (* pIter) -> fileName;
sendBuff += "\"";
sendBuff += "\r\n";
sendBuff += "\r\n";
Send(sendBuff);
ifstream ifs((* pIter) -> filePath, ios::in | ios::binary);
if (false == ifs.is_open())
{
    return 4;                /* 错误码 4 表示文件打开错误 */
}
char fileBuff[MAX_FILE_LEN];
char * chSendBuff;
memset(fileBuff, 0, sizeof(fileBuff));
/* 文件使用 base64 加密传送 */
while (ifs.read(fileBuff, MAX_FILE_LEN))
{
    //cout << ifs.gcount() << endl;
    chSendBuff = base64Encode(fileBuff, MAX_FILE_LEN);
    chSendBuff[strlen(chSendBuff)] = '\r';
    chSendBuff[strlen(chSendBuff)] = '\n';
    send(sockClient, chSendBuff, strlen(chSendBuff), 0);
    delete[] chSendBuff;
}
//cout << ifs.gcount() << endl;
chSendBuff = base64Encode(fileBuff, ifs.gcount());
chSendBuff[strlen(chSendBuff)] = '\r';
chSendBuff[strlen(chSendBuff)] = '\n';
int err = send(sockClient, chSendBuff, strlen(chSendBuff), 0);

if (err != strlen(chSendBuff))
{
    cout << "文件传送出错!" << endl;
    return 1;
}
delete[] chSendBuff;
}
return 0;
```

```

}

bool CSntp::SendEnd()                /* 发送结尾信息 */
{
    string sendBuff;
    sendBuff = "--qwertyuiop--";
    sendBuff += "\r\n.\r\n";
    if (false == Send(sendBuff) || false == Recv())
    {
        return false;
    }
    cout << buff << endl;
    sendBuff.empty();
    sendBuff = "QUIT\r\n";
    return (Send(sendBuff) && Recv());
}

int CSntp::SendEmail_Ex()
{
    if (false == CreateConn())
    {
        return 1;
    }
    //Recv();
    int err = Login();                //先登录
    if (err != 0)
    {
        return err;                  //错误代码必须要返回
    }
    if (false == SendEmailHead())     //发送 EMAIL 头部信息
    {
        return 1;                    /* 错误码 1 是由于网络的错误 */
    }
    if (false == SendTextBody())
    {
        return 1;
    }
    err = SendAttachment_Ex();
    if (err != 0)
    {
        return err;
    }
    if (false == SendEnd())
    {
        return 1;
    }
    return 0;                         /* 0 表示没有出错 */
}

void CSntp::AddAttachment(string &filePath)    //添加附件

```

```
{
FILEINFO * pFile = new FILEINFO;
strcpy(pFile->filePath, filePath.c_str());
const char * p = filePath.c_str();
strcpy(pFile->fileName, p + filePath.find_last_of("\\") + 1);
listFile.push_back(pFile);
}

void CSntp::DeleteAttachment(string &filePath)    //删除附件
{
list<FILEINFO *>::iterator pIter;
for (pIter = listFile.begin(); pIter != listFile.end(); pIter++)
{
    if (strcmp(( * pIter)->filePath, filePath.c_str()) == 0)
    {
        FILEINFO * p = * pIter;
        listFile.remove( * pIter);
        delete p;
        break;
    }
}
}

void CSntp::DeleteAllAttachment()                /* 删除所有的文件 */
{
for (list<FILEINFO *>::iterator pIter = listFile.begin(); pIter != listFile.end();)
{
    FILEINFO * p = * pIter;
    pIter = listFile.erase(pIter);
    delete p;
}
}

void CSntp::SetSrvDomain(string &domain)
{
this->domain = domain;
}

void CSntp::SetUserName(string &user)
{
this->user = user;
}

void CSntp::SetPass(string &pass)
{
this->pass = pass;
}

void CSntp::SetTargetEmail(string &targetAddr)
{
this->targetAddr = targetAddr;
}
```

```

}
void CSmtp::SetEmailTitle(string &title)
{
    this->title = title;
}
void CSmtp::SetContent(string &content)
{
    this->content = content;
}
void CSmtp::SetPort(int port)
{
    this->port = port;
}

/* main.cpp */
#include <stdlib.h>
#include "smtp.h"
#include <iostream>
#include <stdio.h>
#include <string>
#include <string.h>
#include <conio.h>
#include <openssl/md5.h>
#include <openssl/aes.h>
#include <openssl/evp.h>
#define EVP_DES_CBC EVP_des_cbc()
#define MAX_CHAR_SIZE 204800
using namespace std;

unsigned char * encrypt_text(unsigned char * iv, unsigned char * key, unsigned char *
plaintext, int * ciphertext_len, unsigned char * ciphertext)
{
    EVP_CIPHER_CTX en;
    EVP_CIPHER_CTX_init(&en);
    const EVP_CIPHER * cipher_type;
    int input_len = 0;

    cipher_type = EVP_DES_CBC;

    //init cipher
    EVP_EncryptInit_ex(&en, cipher_type, NULL, key, iv);

    const char * p = (const char * )(char * )plaintext;           //转换

    input_len = strlen(p);
    /* allows reusing of 'e' for multiple encryption cycles */

    if (!EVP_EncryptInit_ex(&en, NULL, NULL, NULL, NULL)){
        printf("ERROR in EVP_EncryptInit_ex \n");
    }

```

```

        return NULL;
    }

    int bytes_written = 0;
    //encrypt
    if (!EVP_EncryptUpdate(&en,
        ciphertext, &bytes_written,
        (unsigned char *)plaintext, input_len)) {
        return NULL;
    }
    * ciphertext_len += bytes_written;

    //do padding
    if (!EVP_EncryptFinal_ex(&en,
        ciphertext + bytes_written,
        &bytes_written)){
        printf("ERROR in EVP_EncryptFinal_ex \n");
        return NULL;
    }
    * ciphertext_len += bytes_written;

    EVP_CIPHER_CTX_cleanup(&en);

    return ciphertext;
}

char * MD5(string str)                                //计算 MD5 的值
{
    char data[204800];
    strcpy(data, str.c_str());
    unsigned char md[16] = { 0 };

    MD5_CTX ctx;
    MD5_Init(&ctx);
    MD5_Update(&ctx, data, strlen(data));
    MD5_Final(md, &ctx);
    int i = 0;
    char buf[33] = { 0 };
    char tmp[3] = { 0 };
    for (i = 0; i < 16; i++)
    {
        sprintf(tmp, "%02X", md[i]);
        strcat(buf, tmp);
    }
    return buf;
}

void convert(char * target, char * source, int n)      //转换定长字符
{
    int a;
    char fstr[20];

```

```

a = atoi(source);
sprintf(fstr, "% %0%dd", n);
sprintf(target, fstr, a);
}

int main()
{
string l1, l2, l3, l4, l5, l6;
cout << "----- 发送邮件 -----" << endl;
cout << "请输入发送邮件使用的用户名:";
cin >> l1;

cout << "请输入用户名密码:";
char ch;
while ((ch = _getch()) != 13)
{
    l2 += ch;                                //string 对象重载了 +=
    cout << " * ";
}
cout << endl;
cout << "请输入收件人邮箱:";
cin >> l3;
cout << "请输入邮件主题:";
cin >> l4;
cout << "请输入邮件正文:";
cin >> l5;
//账号信息
l6 = "smtp.163.com";
string MD5val = MD5(l5);
//加密信息
char * m = (char *)l5.c_str();
unsigned char * in = (unsigned char *)m;
unsigned char * out = NULL;
char * iv = "wereachsuccessalready";    //初始化向量
unsigned char * i = (unsigned char *)iv;
char * key = "wearethemoststronger";    //密钥
unsigned char * k = (unsigned char *)key;
int ciphertext_len = 0;
unsigned char ciphertext[MAX_CHAR_SIZE];
unsigned char plaintext[MAX_CHAR_SIZE];
out = encrypt_text(i, k, in, &ciphertext_len, ciphertext);

out[ciphertext_len] = '\0';
l5 = (char *)out;

int l5len;                                //输入的消息内容变为密文后的长度
l5len = l5.length();
l5 += MD5val;                             //把输入的消息的 MD5 的值添加到发送的 content 中

char * l5temp = new char[];
_itoa(l5len, l5temp, 10);
convert(l5temp, l5temp, 3);

```

```

15 += 15temp;                //把 content 长度添加到发送内容的后面

CSmtp smtp(
    25,                        /* smtp 端口 */
    16,                        /* smtp 服务器地址 */
    11,                        /* 你的邮箱地址 */
    12,                        /* 邮箱密码 */
    13,                        /* 目的邮箱地址 */
    14,                        /* 主题 */
    15                         /* 邮件正文 */
);

int err;
if ((err = smtp.SendEmail_Ex()) != 0)
{
    if (err == 1)
        cout << "错误 1: 由于网络不畅通, 发送失败!" << endl;
    if (err == 2)
        cout << "错误 2: 用户名错误, 请核对!" << endl;
    if (err == 3)
        cout << "错误 3: 用户密码错误, 请核对!" << endl;
    if (err == 4)
        cout << "错误 4: 请检查附件目录是否正确, 以及文件是否存在!" << endl;
}
system("pause");
return 0;
return 0;
}

```

## 2) 接收邮件

```

/* pop3.h */
#include <WinSock2.h>
#pragma comment(lib, "ws2_32.lib")    /* 连接 ws2_32.lib 动态链接库 */

class CPop3 {

public:
    CPop3();
    ~CPop3();

    //初始化 POP3 的属性
    bool Create(const char * username, const char * userpwd, const char * svraddr,
        unsigned short port = 110);

    //连接 POP3 服务器
    bool Connect();
    //登录的服务器
    bool Login();
    //利用 list 命令得到所有的邮件数目
    bool List(int& sum);
    //获得序号为 num 的邮件

```

```

bool FetchEx(int num = 1);
//退出命令
bool Quit();

protected:
int GetMailSum(char * buf);

SOCKET m_sock;
char m_username[32];           /* 用户名 */
char m_userpwd[32];           /* 密码 */
char m_svraddr[32];           /* 服务器域名 */
unsigned short m_port;
private:
int Pop3Recv(char * buf, int len, int flags = 0);
};

/* pop3.cpp */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <iostream>
#include "pop3.h"
#include <openssl/md5.h>
#include <openssl/aes.h>
#include <openssl/evp.h>
#define EVP_DES_CBC EVP_des_cbc()
#define MAX_CHAR_SIZE 204800
using namespace std;
//初始化
unsigned char * decrypt_text(unsigned char * iv, unsigned char * key, unsigned char *
ciphertext, int * ciphertext_len, unsigned char * plaintext) {

EVP_CIPHER_CTX de;
EVP_CIPHER_CTX_init(&de);
const EVP_CIPHER * cipher_type;

int bytes_written = 0;
int update_len = 0;
cipher_type = EVP_DES_CBC;
EVP_DecryptInit_ex(&de, cipher_type, NULL, key, iv);

if (!EVP_DecryptInit_ex(&de, NULL, NULL, NULL, NULL)){
    printf("ERROR in EVP_DecryptInit_ex\n");
    return NULL;
}
int plaintext_len = 0;
if (!EVP_DecryptUpdate(&de,
    plaintext, &update_len,
    ciphertext, * ciphertext_len)){
    printf("ERROR in EVP_DecryptUpdate\n");

```

```

        return NULL;
    }

    if (!EVP_DecryptFinal_ex(&de,
        plaintext + update_len, &bytes_written)){
        printf("ERROR in EVP_DecryptFinal_ex\n");
        return NULL;
    }
    bytes_written += update_len;
    * (plaintext + bytes_written) = '\0';

    EVP_CIPHER_CTX_cleanup(&de);

    return plaintext;
}

char * MD5(string str)
{
    char data[204800];
    strcpy(data, str.c_str());
    unsigned char md[16] = { 0 };

    MD5_CTX ctx;
    MD5_Init(&ctx);
    MD5_Update(&ctx, data, strlen(data));
    MD5_Final(md, &ctx);
    int i = 0;
    char buf[33] = { 0 };
    char tmp[3] = { 0 };
    for (i = 0; i < 16; i++)
    {
        sprintf(tmp, "%02X", md[i]);
        strcat(buf, tmp);
    }
    return buf;
}

CPop3::CPop3()
{
    WSADATA wsaData;
    WORD version = MAKEWORD(2, 0);
    WSAStartup(version, &wsaData);
}

CPop3::~~CPop3()
{
    WSACleanup();
}

```

```

int CPop3::Pop3Recv(char * buf, int len, int flags)
{
    /* 接收数据 */
    int rs;
    int offset = 0;
    do
    {
        if (offset > len - 2)
            return offset;

        rs = recv(m_sock, buf + offset, len - offset, flags);
        if (rs < 0) return -1;
        offset += rs;
        buf[offset] = '\0';
    } while (strstr(buf, "\r\n.\r\n") == (char *)NULL);

    return offset;
}

bool CPop3::Create(
    const char * username,          //用户名
    const char * userpwd,          //用户密码
    const char * svraddr,          //服务器地址
    unsigned short port            //服务端口
)
{
    strcpy(m_username, username);
    strcpy(m_userpwd, userpwd);
    strcpy(m_svraddr, svraddr);
    m_port = port;

    return true;
}

bool CPop3::Connect()
{
    //创建套接字
    m_sock = socket(AF_INET, SOCK_STREAM, 0);
    //IP 地址
    char ipaddr[16];

    struct hostent * p;
    if ((p = gethostbyname(m_svraddr)) == NULL) //如果得不到服务器信息,就说明出错
    {
        return FALSE;
    }

    sprintf(
        ipaddr,
        "%u.%u.%u.%u",
        (unsigned char)p->h_addr_list[0][0],

```

```

        (unsigned char)p->h_addr_list[0][1],
        (unsigned char)p->h_addr_list[0][2],
        (unsigned char)p->h_addr_list[0][3]
    );

//连接 POP 服务器
struct sockaddr_in svraddr;
svraddr.sin_family = AF_INET;
svraddr.sin_addr.s_addr = inet_addr(ipaddr);
svraddr.sin_port = htons(m_port);
int ret = connect(m_sock, (struct sockaddr*)&svraddr, sizeof(svraddr));
if (ret == SOCKET_ERROR)
{
    return FALSE;
}

//接收 POP3 服务器发来的欢迎信息
char buf[128];
int rs = recv(m_sock, buf, sizeof(buf), 0);
buf[rs] = '\0';

printf("%s", buf);
if (rs <= 0 || strncmp(buf, "+OK", 3) != 0) /* 服务器没有返回 OK 就出错了 */
{
    return FALSE;
}

return TRUE;
}

bool CPop3::Login()
{
    /* 登录 */

    /* 发送用户命令 */
    char sendbuf[128];
    char recvbuf[128];

    sprintf(sendbuf, "USER %s\r\n", m_username);
    printf("%s", sendbuf);
    send(m_sock, sendbuf, strlen(sendbuf), 0); /* 发送用户名 */

    int rs = recv(m_sock, recvbuf, sizeof(recvbuf), 0); //接收服务器发来的信息
    recvbuf[rs] = '\0';
    if (rs <= 0 || strncmp(recvbuf, "+OK", 3) != 0) /* 如果没有 "+OK" 就说明失败了 */
    {
        return FALSE;
    }
    printf("%s", recvbuf);
    /* 发送密码信息 */
    memset(sendbuf, 0, sizeof(sendbuf));

```

```

sprintf(sendbuf, "PASS %s\r\n", m_userpwd);
send(m_sock, sendbuf, strlen(sendbuf), 0);
printf("%s", sendbuf);

rs = recv(m_sock, recvbuf, sizeof(recvbuf), 0);
recvbuf[rs] = '\0';
if (rs <= 0 || strncmp(recvbuf, "+OK", 3) != 0)
{
    return FALSE;
}
printf("%s", recvbuf);
return TRUE;
}

bool CPop3::List(int& sum)
{
    /* 发送 LIST 命令 */
    char sendbuf[128];
    char recvbuf[256];
    sprintf(sendbuf, "LIST\r\n");
    send(m_sock, sendbuf, strlen(sendbuf), 0);
    printf("%s", sendbuf);

    int rs = Pop3Recv(recvbuf, sizeof(recvbuf), 0);
    if (rs <= 0 || strncmp(recvbuf, "+OK", 3) != 0)
    {
        return FALSE;
    }
    recvbuf[rs] = '\0';
    printf("%s", recvbuf);
    sum = GetMailSum(recvbuf);
    cout << sum << endl;
    return TRUE;
}

bool CPop3::FetchEx(int num)
{
    int rs;
    FILE* fp;
    int flag = 0;
    unsigned int len;
    char filename[256];
    char sendbuf[128];
    char recvbuf[20480];
    /* 发送 RETR 命令 */
    sprintf(sendbuf, "RETR %d\r\n", num);
    send(m_sock, sendbuf, strlen(sendbuf), 0);
    do
    {
        rs = Pop3Recv(recvbuf, sizeof(recvbuf), 0);    //接收数据
        if (rs < 0)

```

```

{
    return FALSE;
}
recvbuf[rs] = '\0';
printf("Recv RETR Resp:\n %s", recvbuf);           //输出接收的数据
if (flag == 0)
{
    _itoa(num, filename, 10);                       //按照序号给文件排名
    strcat(filename, ".eml");

    flag = 1;
    fp = fopen(filename, "wb");                     //准备写文件
}
len = strlen(recvbuf);
char * contentlen = new char[];                   //邮件内容信息的长度
int contentlength;
char md5val[33];                                   //取出的 MD5 的值
md5val[32] = '\0';                                //终结符
memcpy(contentlen, recvbuf + len - 23 - 3, 3);     //取消息长度
memcpy(md5val, recvbuf + len - 23 - 3 - 32, 32);   //取哈希值
contentlength = atoi(contentlen);
char * content = new char[];
memcpy(content, recvbuf + len - 23 - 3 - 32 - contentlength, contentlength); //取内容
content[contentlength] = '\0';                     //把内容补充成为字符串

unsigned char * final = NULL;
char * iv = "wereachsuccessalready";
unsigned char * i = (unsigned char *)iv;
char * key = "wearethemoststronger";
unsigned char * k = (unsigned char *)key;
int ciphertext_len = 0;
unsigned char ciphertext[MAX_CHAR_SIZE];
unsigned char plaintext[MAX_CHAR_SIZE];
final = decrypt_text(i, k, (unsigned char *)content, &contentlength, plaintext);
content = (char *)final;
content[strlen(content)] = '\0';
if (strcmp(MD5(content), md5val) == 0)
    cout << "邮件内容完整性良好!" << endl;
fwrite(recvbuf, 1, len - 23 - 3 - 32 - contentlength, fp);
fwrite(content, 1, strlen(content), fp);
fwrite(recvbuf + len - 23, 1, 24, fp);
fflush(fp);                                         //刷新
} while (strstr(recvbuf, "\r\n.\r\n") == (char *)NULL);
fclose(fp);
return TRUE;
}
bool CPop3::Quit()
{
    char sendbuf[128];

```

```
char recvbuf[128];

/* 发送 QUIT 命令 */
sprintf(sendbuf, "QUIT\r\n");
send(m_sock, sendbuf, strlen(sendbuf), 0);
int rs = recv(m_sock, recvbuf, sizeof(recvbuf), 0);
if (rs <= 0 || strncmp(recvbuf, "+OK", 3) != 0)
{
    return FALSE;
}
closesocket(m_sock);
return TRUE;
}
```

```
int CPop3::GetMailSum(char * buf)
{
    int sum = 0;
    char * p = strstr(buf, "\r\n");
    if (p == NULL)
        return sum;
    p = strstr(p + 2, "\r\n");
    if (p == NULL)
        return sum;
    while ((p = strstr(p + 2, "\r\n")) != NULL)
    {
        sum++;
    }

    return sum;
}
```

```
/* main.cpp */
#include <stdio.h>
#include "pop3.h"
#include <iostream>
#include <string>
#include <string.h>
#include <conio.h>
#include <iostream>
#include <stdlib.h>
#include <openssl/md5.h>

using namespace std;
int main()
```

```
{
    * /
    cout << " ----- 登入邮箱 ----- " << endl;
    int i;
    int sum;
    string l1, l2, l3;
    CPop3 pop3;
    cout << "请输入服务器地址(pop):";
    cin >> l3;
    cout << "请输入用户名:";
    cin >> l1;
    cout << "请输入密码:";
    char ch;
    while ((ch = _getch()) != 13)
    {
        l2 += ch;                //string 对象重载了 +=
        cout << " * ";
    }
    cout << endl;
    char userName[256];
    char password[256];
    char srv[256];
    for (i = 0; i < l3.length(); i++){
        srv[i] = l3[i];
    }
    srv[i] = '\0';
    for (i = 0; i < l2.length(); i++){
        password[i] = l2[i];
    }
    password[i] = '\0';
    for (i = 0; i < l1.length(); i++){
        userName[i] = l1[i];
    }
    userName[i] = '\0';
    pop3.Create(userName, password, srv, 110);
    pop3.Connect();                //连接 pop3 服务器
    pop3.Login();
    pop3.List(sum);
    if (sum < 0)
        printf("You have no letter in box !");
    int n;
    cout << "一共有" << sum << "封邮件, 请输入你想查看第几封邮件: ";
    cin >> n;
    pop3.FetchEx(n);
    pop3.Quit();

    return 0;
}
```

运行结果如下。

1) 发送邮件(图 8-6 与图 8-7)

```

C:\Windows\system32\cmd.exe
-----发送邮件-----
请输入发送邮件使用的用户名: zxc-1267@163.com
请输入用户名密码: *****
请输入收件人邮箱: 163.com
请输入邮件主题: hello
请输入邮件正文: 你好
220 163.com Anti-spam GT for Coremail System <163com[20141201]>

EHLO zxc-1267@163.com

250-mail
250-PIPELINING
250-AUTH LOGIN PLAIN
250-AUTH=LOGIN PLAIN
250-coremail 1Uxr2xKj7kC0xkl17xCrU7I0s8FY2U3Uj8Ca2Bx1U0000U7Ic2I0Y2Uftd-eSUCa0xD...
250-STARTTLS
250 8BITMIME

AUTH LOGIN

334 dXN1cm5hbWU6

enhjLTEgMjc=
搜狗拼音输入法 全 :

```

图 8-6 邮件发送页面

```

C:\Windows\system32\cmd.exe

From: zxc-1267@163.com
To: 163.com
Subject: hello
MIME-Version: 1.0
Content-Type: multipart/mixed;boundary=qwertyuiop

---qwertyuiop
Content-Type: text/plain;charset="gb2312"
755755B94AE3C6D892B29CF48D9BEA819B27B9008

---qwertyuiop---
.

250 Mail OK queued as smtp4,DtGowECZeUSgUH1WkMdaA--.2701S2 1450792097
250 Mail OK queued as smtp4,DtGowECZeUSgUH1WkMdaA--.2701S2 1450792097

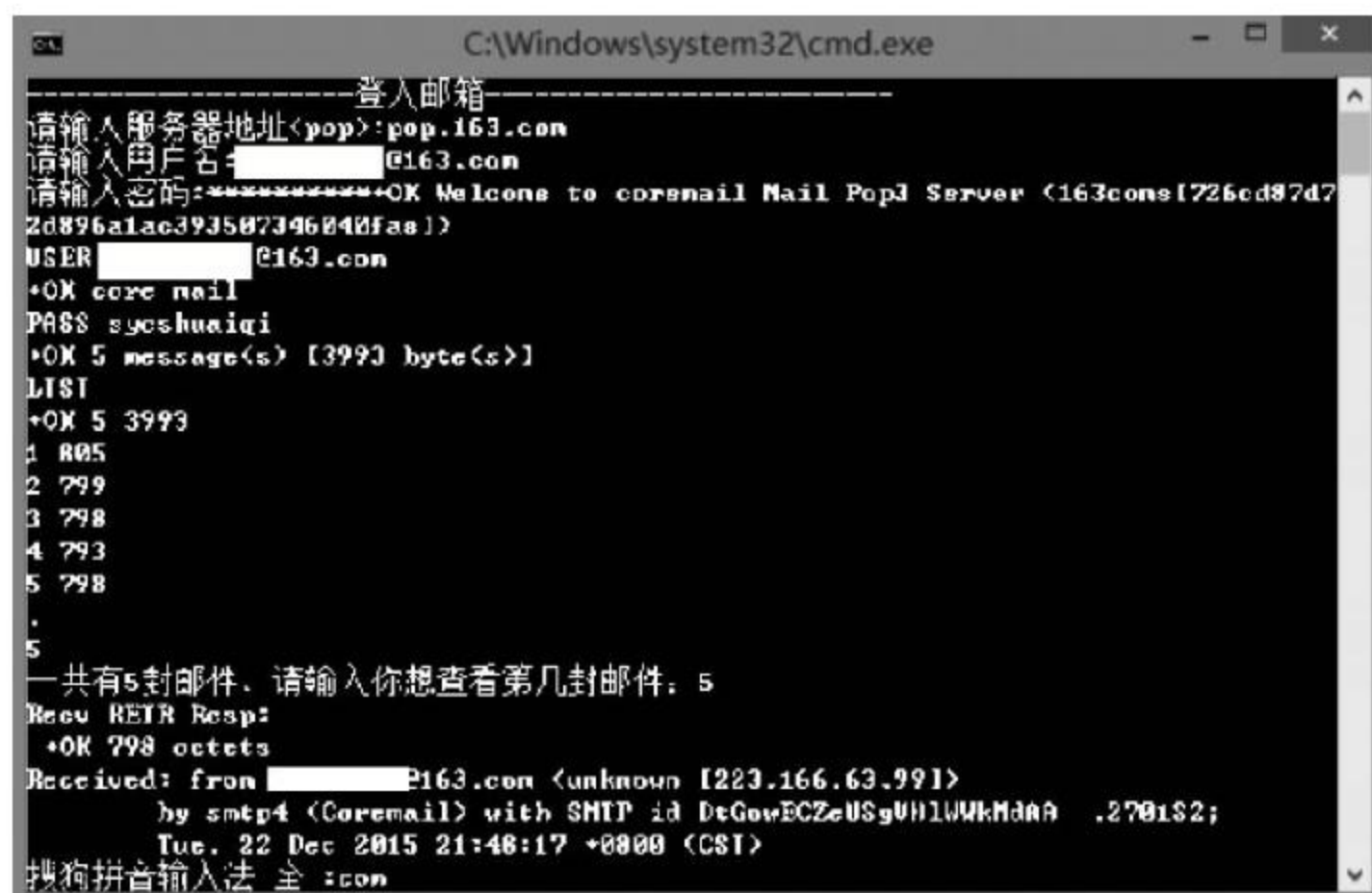
QUIT

221 Bye
搜狗拼音输入法 全 :

```

图 8-7 邮件发送成功

## 2) 接收邮件(图 8-8~图 8-10)

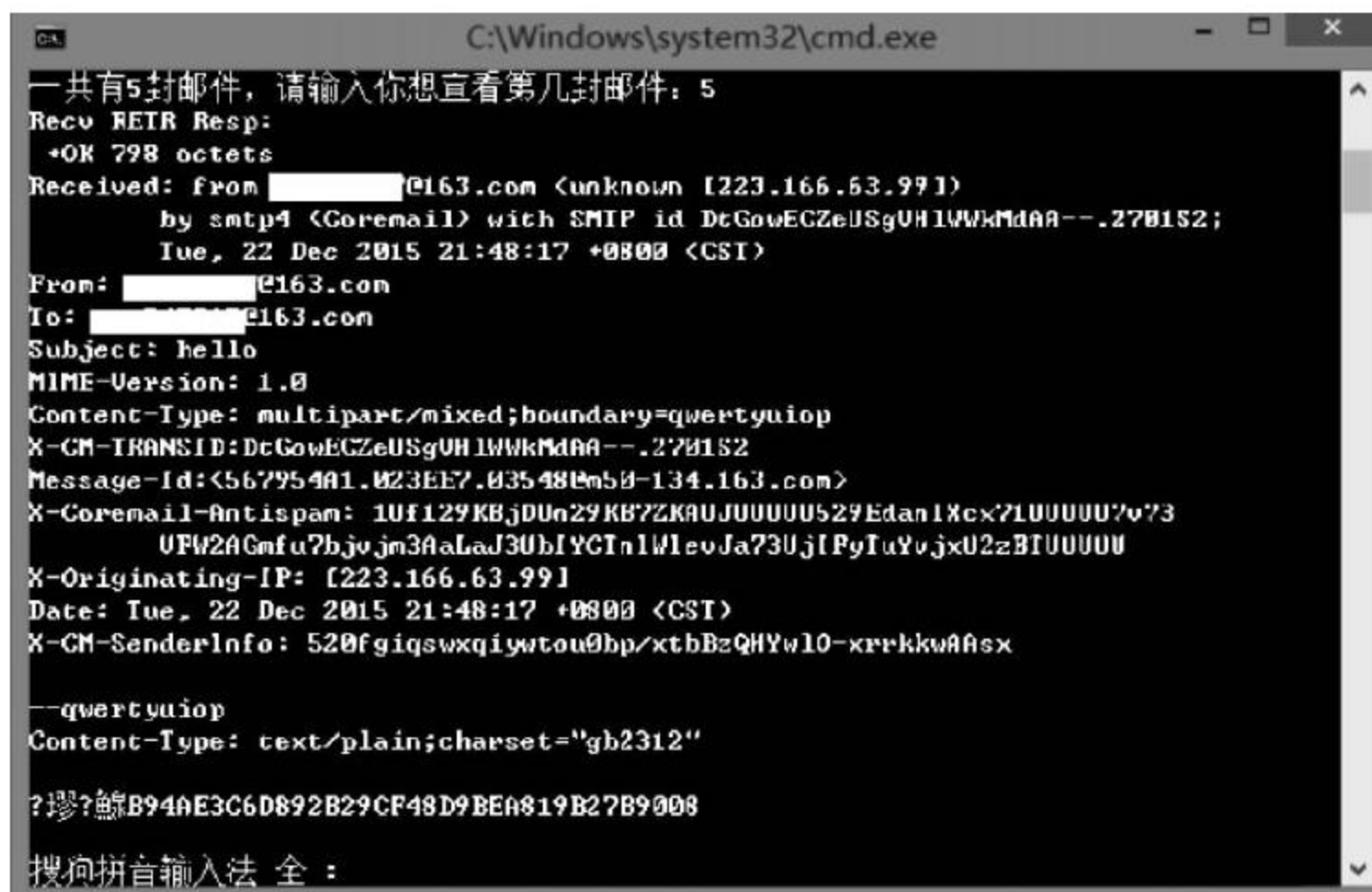


```

C:\Windows\system32\cmd.exe
-----登入邮箱-----
请输入服务器地址(pop):pop.163.com
请输入用户名: @163.com
请输入密码:*****OK Welcome to coremail Mail Pop3 Server (163cons1726cd87d7
2d896a1ac393587346840fas)
USER @163.com
+OK core mail
PASS syeshuaiqi
+OK 5 message(s) [3993 byte(s)]
LIST
+OK 5 3993
1 805
2 799
3 798
4 793
5 798
.
5
一共有5封邮件, 请输入你想查看第几封邮件: 5
Recv RETR Resp:
+OK 798 octets
Received: from @163.com (unknown [223.166.63.99])
        by smtp4 (Coremail) with SMTP id DtGowECZeUSgUHLWWkMdaA .2701S2;
        Tue, 22 Dec 2015 21:48:17 +0800 (CST)
搜狗拼音输入法 全 :con

```

图 8-8 邮件接收页面



```

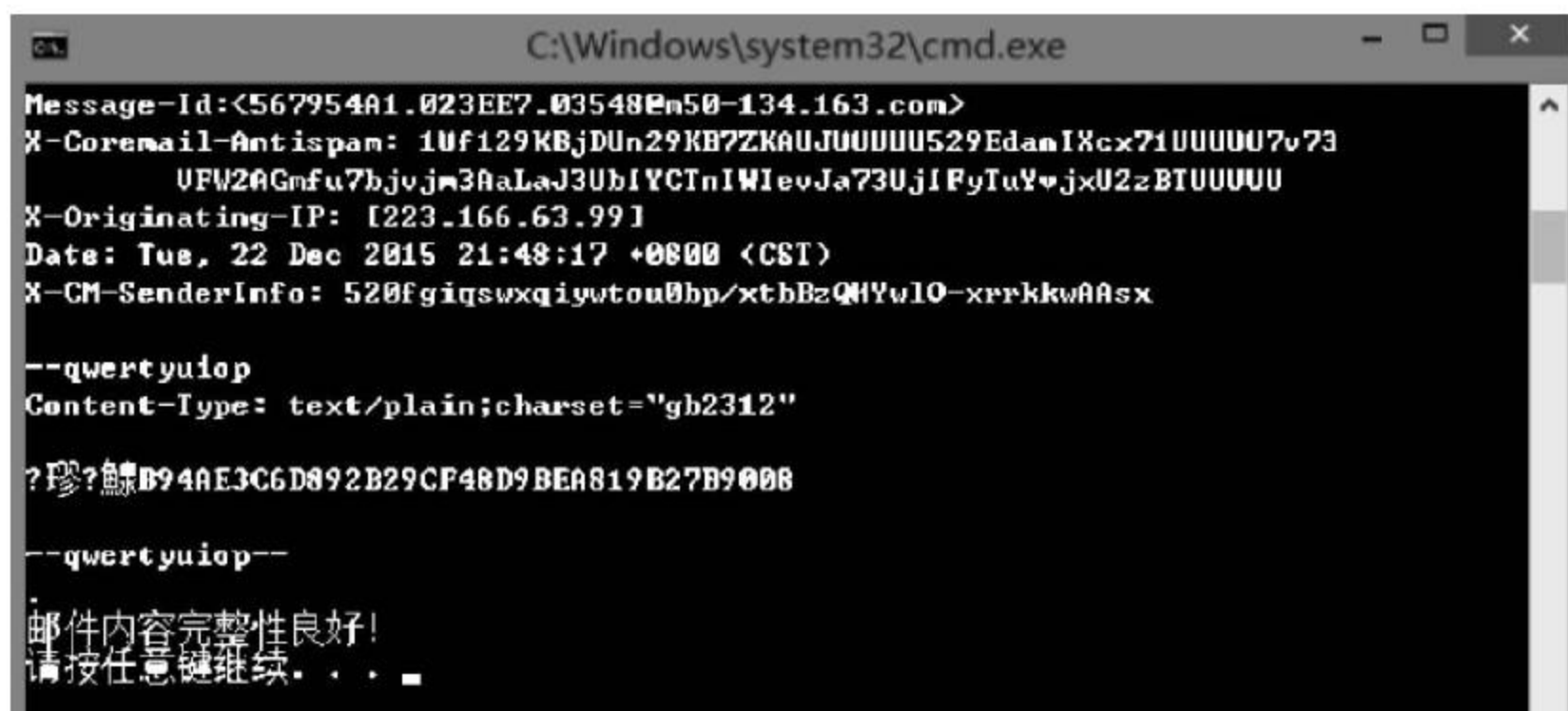
C:\Windows\system32\cmd.exe
一共有5封邮件, 请输入你想查看第几封邮件: 5
Recv RETR Resp:
+OK 798 octets
Received: from @163.com (unknown [223.166.63.99])
        by smtp4 (Coremail) with SMTP id DtGowECZeUSgUHLWWkMdaA .2701S2;
        Tue, 22 Dec 2015 21:48:17 +0800 (CST)
From: @163.com
To: @163.com
Subject: hello
MIME-Version: 1.0
Content-Type: multipart/mixed;boundary=quertyuiop
X-CM-TRANSID:DtGowECZeUSgUHLWWkMdaA .2701S2
Message-Id:<567954A1.023EE7.03548E50-134.163.com>
X-Coremail-Antispam: 1Uf129KBjDUn29KB7ZKAUJU0000U529EdanIXcx71U0000U7v73
UFW2AGmfu7bjvjm3AaLaJ3UbiYCTnIWleuJa73UjIFyTuYvjxU2zBTU0000
X-Originating-IP: [223.166.63.99]
Date: Tue, 22 Dec 2015 21:48:17 +0800 (CST)
X-CM-SenderInfo: 520fgiqswxqiywtou0bp/xtbBzQHw10-xrrkkwAAsx

--quertyuiop
Content-Type: text/plain;charset="gb2312"

?珍?鯨B94AE3C6D892B29CF48D9BEA819B27B9008
搜狗拼音输入法 全 :

```

图 8-9 邮件接收过程



```

C:\Windows\system32\cmd.exe
Message-Id:<567954A1.023EE7.03548E50-134.163.com>
X-Coremail-Antispam: 1Uf129KBjDUn29KB7ZKAUJU0000U529EdanIXcx71U0000U7v73
UFW2AGmfu7bjvjm3AaLaJ3UbiYCTnIWleuJa73UjIFyTuYvjxU2zBTU0000
X-Originating-IP: [223.166.63.99]
Date: Tue, 22 Dec 2015 21:48:17 +0800 (CST)
X-CM-SenderInfo: 520fgiqswxqiywtou0bp/xtbBzQHw10-xrrkkwAAsx

--quertyuiop
Content-Type: text/plain;charset="gb2312"

?珍?鯨B94AE3C6D892B29CF48D9BEA819B27B9008

--quertyuiop--
邮件内容完整性良好!
请按任意键继续. . .

```

图 8-10 邮件接收成功

## 小 结

本章主要列举了安全编程的几个常见的应用程序,即基于 OpenSSL 的安全 Web 服务器和安全电子邮件传输。对各个应用程序的相关基础知识分别进行了简要介绍,并提供了相应的实现代码,为读者进一步加深理解安全编程提供良好的基础。

## 思 考 题

1. SSL 协议是如何工作的?
2. HTTPS 与 HTTP 的区别是什么?
3. 描述电子邮件的传输过程。
4. 思考 Web 服务器安全程序还有哪些实现方式,并尝试实现其中一种。
5. 安全电子邮件编程中如何实现对垃圾邮件的过滤? 请尝试实现。

## 参考文献

- [1] 李晖,张宁,等. 网络空间安全学科人才培养之思考[J]. 网络与信息安全学报,2015,(1).
- [2] 李建华,邱卫东,孟魁,等. 网络空间安全一级学科内涵建设和人才培养思考[J]. 信息安全研究,2015.
- [3] 崔光耀,冯雪竹,等. 强力推进网络空间安全一级学科建设——访沈昌祥院士[J]. 中国信息安全,2015.
- [4] 李红娇,等. 信息安全概论[M]. 2 版. 北京: 中国电力出版社,2016.
- [5] [美]William Stallings,[澳] Lawrie Brown. 计算机安全——原理与实践[M]. 2 版. 王昭,等译. 北京: 电子工业出版社,2015.
- [6] [美]William Stallings. 密码编码学与网络安全——原理与实践[M]. 6 版. 唐明,等译. 北京: 电子工业出版社,2015.
- [7] 陈卓,阮鸥,沈剑. 网络安全编程与实践[M]. 北京: 国防工业出版社,2008.
- [8] 刘文涛. 网络安全编程技术与实例[M]. 北京: 机械工业出版社,2008.
- [9] 吴功宜,张健忠,张健,等. 网络安全高级软件编程技术[M]. 北京: 清华大学出版社,2010.
- [10] 李峰,陈向益,等. TCP/IP 协议分析与应用编程[M]. 北京: 人民邮电出版社,2008.
- [11] 任泰明. TCP/IP 网络编程[M]. 北京: 人民邮电出版社,2009.
- [12] Ofir Arkin. ICMP usage in Scanning[R/OL]. [http://www.sys-security.com/architect/papers/icmp\\_scanning\\_v3.0.pdf](http://www.sys-security.com/architect/papers/icmp_scanning_v3.0.pdf).
- [13] Fyodor. Remote OS Detection via TCP/IP Fingerprinting (2nd Generation) [R/OL]. <http://nmap.org/osdetect/>.
- [14] ICMP TYPE NUMBERS[R/OL]. <http://www.iana.org/assignments/icmp-parameters>.
- [15] 连一峰,戴英侠,胡艳,等. 分布式入侵检测模型研究[J]. 计算机研究与发展,2003,40(8).
- [16] 刘春. 基于组合算法选择特征的网络入侵检测模型[J]. 计算机与现代化,2014,(8).
- [17] 李玉霞,刘丽,沈桂兰. 基于 AFSA-SVM 的网络入侵检测模型[J]. 计算机工程与应用,2013,49(24).
- [18] 魏群,刘玉芳,刘保相. 电子商务中 CA 认证的 OPENSSL 实现方法[J]. 微计算机信息,2008,24(6).
- [19] 刘维岗. 基于 IPv6 的防火墙包过滤技术研究[J]. 计算机与现代化,2012,(3).

## 网络链接

- [1] [http://news.xinhuanet.com/politics/2015-06/04/c\\_127879542.htm](http://news.xinhuanet.com/politics/2015-06/04/c_127879542.htm),积极构建网络空间安全创新人才培养体系.
- [2] <http://www.myhack58.com/Article/html/3/68/2013/39849.html>.
- [3] [http://xidong.net/File001/File\\_75831.html](http://xidong.net/File001/File_75831.html).
- [4] <http://blog.csdn.net/column/details/visualcppsafe.html>.
- [5] [http://baike.baidu.com/link?url=Z\\_22\\_GylROZRUIPeSvPwik7nXPJIIJOAF3mhjQsYt2UQ5KPRyAF-FrhWfpM4A1zl0yT7pcmc8PP9nZgjAqcw1Rq](http://baike.baidu.com/link?url=Z_22_GylROZRUIPeSvPwik7nXPJIIJOAF3mhjQsYt2UQ5KPRyAF-FrhWfpM4A1zl0yT7pcmc8PP9nZgjAqcw1Rq).
- [6] <http://blog.csdn.net/gdwzh/article/details/19230>.

## 图书资源支持

感谢您一直以来对清华版图书的支持和爱护。为了配合本书的使用,本书提供配套的素材,有需求的用户请到清华大学出版社主页(<http://www.tup.com.cn>)上查询和下载,也可以拨打电话或发送电子邮件咨询。

如果您在使用本书的过程中遇到了什么问题,或者有相关图书出版计划,也请您发邮件告诉我们,以便我们更好地为您服务。

### 我们的联系方式:

地 址: 北京海淀区双清路学研大厦 A 座 707

邮 编: 100084

电 话: 010-62770175-4604

资源下载: <http://www.tup.com.cn>

电子邮件: [weijj@tup.tsinghua.edu.cn](mailto:weijj@tup.tsinghua.edu.cn)

QQ: 883604 (请写明您的单位和姓名)

用微信扫一扫右边的二维码,即可关注清华大学出版社公众号“书圈”。



扫一扫

资源下载、样书申请  
新书推荐、技术交流